AMENDMENTS TO THE CLAIMS

1. (Currently Amended) A method for rendering a three-dimensional (3D) graphical image, said method comprising the steps of:

representing said 3D graphical image as a plurality of graphics primitives each having a plurality of vertices;

for each of said plurality of graphics primitives, computing at least two texture coordinate gradient vectors, wherein said computing comprises computing a first dot product (W_{00}) of a first vector (D_1) with itself;

for each vertex of said plurality of graphics primitives, determining a 3D coordinate frame, wherein said determining step includes using said at least two texture coordinate gradient vectors computed for the respective graphics primitive for orienting said 3D coordinate frame; and

utilizing at least said 3D coordinate frame to determine parameters of a parametric texture mapping function.

2. (Original) The method of claim 1 further comprising the step of:

evaluating said parametric texture mapping function for rendering said 3D graphical image.

3. (Original) The method of claim 2 wherein said parametric texture mapping function comprises a biquadric polynomial having at least six coefficients.

4. (Original) The method of claim 1 wherein said step of utilizing comprises:

calculating scalar components for said parametric texture mapping function.

5. (Original) The method of claim 4 wherein said scalar components include lighting scalar components.

6. (Original) The method of claim 1 wherein said 3D coordinate frame is formed by a normal vector, tangent vector, and binormal vector.

7. (Original) The method of claim 6 wherein said step of utilizing further comprises:

calculating a first lighting scalar component for said parametric texture mapping

function as the dot product of a light vector and said tangent vector; and

calculating a second lighting scalar component for said parametric texture mapping function as the dot product of said light vector and said binormal vector.

8. (Original) The method of claim 1 wherein said step of computing at least two texture coordinate gradient vectors comprises:

computing a first texture coordinate gradient vector that identifies the direction of maximum change along a first texture coordinate; and

computing a second texture coordinate gradient vector that identifies the direction of maximum change along a second texture coordinate.

9. (Original) The method of claim 1 wherein said step of computing at least two texture coordinate gradient vectors comprises:

assigning a computed value to a variable; and

utilizing said variable in computing each of said at least two texture coordinate gradient vectors.

10. (Original) The method of claim 9 wherein said utilizing step further comprises:

using said variable as a denominator in calculating linear combination variables that are used in further computing said at least two texture coordinate gradient vectors.

11. (Original) The method of claim 1 wherein said parametric texture mapping function is a luminance parametric texture mapping function.

12. (Original) The method of claim 1 wherein said parametric texture mapping function is a red-green-blue (RGB) parametric texture mapping function.

13. (Original) The method of claim 1 wherein said graphics primitive comprises a polygon.

14. (Original) A method for mapping a computer graphics texture to a three-dimensional (3D) object, said method comprising the steps of:

representing said 3D object as a plurality of graphics primitives;

determining a first vector from a first vertex of a graphics primitive to a second vertex of said graphics primitive;

determining a second vector from said first vertex to a third vertex of said graphics primitive;

calculating a first dot product of said first vector by said first vector;

calculating a second dot product of said first vector and said second vector;

assigning one variable a value derived from at least said first dot product and said second dot product;

computing at least two texture coordinate gradient vectors utilizing at least said one variable, wherein said at least two texture coordinate gradient vectors are indicative of orientation of a texture mapped to said graphics primitive;

determining a 3D coordinate frame for each vertex of said graphics primitive, wherein said determining comprises using said at least two texture coordinate gradient vectors for orienting said 3D coordinate frame; and

utilizing at least said 3D coordinate frame in mapping said texture to said 3D object.

15. (Original) The method of claim 14 wherein said mapping said texture to said 3D object comprises utilizing a parametric texture map function.

16. (Previously Presented) A system for rendering three-dimensional (3D) graphical images, said system comprising:

data structure, stored to computer-readable media, representing said 3D graphical image as a plurality of graphics primitives each having a plurality of vertices;

software code, stored to computer-readable media, executable to compute at least two texture coordinate gradient vectors for each of said plurality of graphics primitives;

software code, stored to computer-readable media, executable to determine a 3D coordinate frame for each vertex of said plurality of graphics primitives, wherein said software code uses said at least two texture coordinate gradient vectors computed for the respective graphics primitive for orienting said 3D coordinate frame; and

software code, stored to computer-readable media, executable to utilize at least said 3D coordinate frame to determine parameters of a parametric texture mapping function.

17. (Original) The system of claim 16 wherein said software code executable to compute at least two texture coordinate gradient vectors further comprises:

software code executable to assign a computed value to a variable; and

software code executable to utilize said variable in computing each of aid at least two texture coordinate gradient vectors.

18. (Original) The system of claim 16 wherein said software code executable to compute at least two texture coordinate gradient vectors further comprises:

software code executable to use said variable as a denominator in calculating linear combination variables that are used in further computing said at least two texture coordinate gradient vectors.

19. (Original) The system of claim 16 wherein said 3D coordinate frame is formed by a normal vector, tangent vector, and binormal vector.

20. (Original) The system of claim 19 wherein said software code executable to determine a 3D coordinate frame further comprises:

software code executable to calculate a first lighting scalar component for said parametric texture mapping function as the dot product of a light vector and said tangent vector; and

software code executable to calculate a second lighting scalar component for said parametric texture mapping function as the dot product of said light vector and said binormal vector.

21. (Previously Presented) The method of claim 1 wherein said parametric texture mapping (PTM) function comprises biquadric function $PTM(s,t,l_u,l_v) = A(s,t)l_u^2 + B(s,t)l_v^2 + C(s,t)l_u l_v + D(s,t)l_u + E(s,t)l_v + F(s,t)$, where s and t are texture coordinates, lu and lv are independent variables representing light direction, and A, B, C, D, E, and F are coefficients.

22. (Previously Presented) The method of claim 1 wherein said parametric texture mapping (PTM) function comprises biquadric function $PTM(v_u,v_v,l_u,l_v) = A(v_u,v_v)l_u^2 + B(v_u,v_v)l_v^2 + C(v_u,v_v)l_u l_v + D(v_u,v_v)l_u + E(v_u,v_v)l_v + F(v_u,v_v)$ where v_u and v_v are components of a view vector representing view direction, lu and lv are components of a light vector representing light direction, and A, B, C, D, E, and F are coefficients.

23. (Previously Presented) The method of claim 1 wherein said parametric texture mapping (PTM) function comprises biquadric function $PTM(h_u,h_v,d_u,d_v) = A(h_u,h_v)d_u^2 + B(h_u,h_v)d_v^2 + C(h_u,h_v)d_u d_v + D(h_u,h_v)d_u + E(h_u,h_v)d_v + F(h_u,h_v)$ where h_u and h_v are components of a half-angle vector, du and dv are components of a difference vector, and A, B, C, D, E, and F are coefficients.

24. (Previously Presented) A method for rendering a digital image of an object comprising:

using a plurality of graphics primitives to represent the object, wherein each of the graphics primitives has a plurality of vertices;

for each of the graphics primitives of the digital image, determining

(a) a gradient vector (G_s) that identifies the direction of maximum change in a first texture coordinate (*s*), and

(b) a gradient vector (G_t) that identifies the direction of maximum change in a second texture coordinate (*t*);

determining, for each vertex of at least one of the graphics primitives, a 3D coordinate frame formed by a surface normal vector, tangent vector, and binormal vector, where the tangent vector is determined as directed in the general direction of the *s* texture coordinate gradient vector (G_s) determined for the at least one graphics primitive and where the binormal vector is determined as directed in the general direction of the *t* texture coordinate gradient vector (G_t) determined for the at least one graphics primitive;

using the determined 3D coordinate frame of said at least one graphics primitive for solving a parametric texture mapping (PTM) function for each pixel of the corresponding graphics primitive.

25. (Previously Presented) The method of claim 24 comprising:

determining, for each vertex of each of the graphics primitives, a 3D coordinate frame formed by a surface normal vector, tangent vector, and binormal vector, where the tangent vector is determined as directed in the general direction of the *s* texture coordinate gradient vector (G_s) determined for the corresponding graphics primitive and where the binormal vector is determined as directed in the general direction of the *t* texture coordinate gradient vector (G_t) determined for the corresponding graphics primitive.

26. (Previously Presented) The method of claim 24 where the *t* texture coordinate is orthogonal to the *s* texture coordinate.

27. (Previously Presented) The method of claim 24 where said PTM function comprises biquadric function $PTM(s,t,l_u,l_v) = A(s,t)l_u^2 + B(s,t)l_v^2 + C(s,t)l_u l_v + D(s,t)l_u + E(s,t)l_v + F(s,t)$, where s and t are texture coordinates, *lu* and *lv* are independent variables representing light direction, and A, B, C, D, E, and F are coefficients.

28. (Currently Amended) A method for rendering a three-dimensional (3D) graphical image, said method comprising:

representing said 3D graphical image as a plurality of graphics primitives each having a plurality of vertices;

for each of said plurality of graphics primitives, computing at least two texture coordinate gradient vectors<u>, wherein said computing comprises computing a dot product of a first vector with itself</u>;

for each vertex of said plurality of graphics primitives, determining a 3D coordinate frame, wherein said determining step includes using said at least two texture coordinate gradient vectors computed for the respective graphics primitive for orienting said 3D coordinate frame; and

utilizing at least said 3D coordinate frame to determine parameters of a parametric texture map (PTM) that includes a biquadric function having at least four independent variables, where a first two of said four independent variables are for indexing the PTM and the other two of said four independent variables are for evaluating the PTM function.

29. (Previously Presented) The method of claim 28 wherein the four independent variables comprise s, t, l_u, and l_v, where s and t are texture coordinates and l_u and l_v are components of a light vector representing light direction.

30. (Previously Presented) The method of claim 28 wherein the four independent variables comprise l_u, l_v, v_u, and v_v, where l_u and l_v are components of a light vector representing light direction and where v_u and v_v are components of a view vector representing view direction.

31. (Previously Presented) The method of claim 28 wherein the four independent variables comprise h_u, h_v, d_u, and d_v, where h_u and h_v are components of a half-angle vector and where *du* and *dv* are components of a difference vector.

32. (Previously Presented) A method comprising:

using a plurality of graphics primitives to represent the object, wherein each of the graphics primitives has a plurality of vertices, where at least one of the graphics primitives has a first vertex position (XYZ_0), a second vertex position (XYZ_1), and a third vertex position (XYZ_2);

associating texture coordinates of a texture map with each of the vertices of the at least one graphics primitives, where texture coordinates ST_0, ST_1, and ST_2 are associated with vertices XYZ_0, XYZ_1, and XYZ_2, respectively;

determine a first vector (D_1) from the first vertex position (XYZ_0) to the second vertex position (XYZ_1);

determine a second vector (D_2) from the first vertex position (XYZ_0) to the third vertex position (XYZ_2);

determine $T_1=ST_1-ST_0$;

determine $T_2=ST_2-ST_0$;

compute first dot product (W_{00}) of first vector (D_1) with itself ($W_{00}=D_1 \bullet D_1$);

compute second dot product (W_{01}) of first vector (D_1) with second vector (D_2) ($W_{01} = D_1 \bullet D_2$);

compute third dot product (W_{11}) of second vector (D_2) with itself ($W_{11} = D_2 \bullet D_2$);

compute denom $= 1 / (W_{00} * W_{11} - W_{01} * W_{01})$;

compute $p_s = (T_1s * W_{11} + T_2s * W_{01}) *$ denom;

compute $q_s = (T_2s * W_{00} + T_1S * W_{01}) *$ denom;

determine a first gradient vector (G_s) that identifies the direction of maximum change in a first texture coordinate (s) as $G_s = p_s * D_1 + q_s * D_2$;

compute $p_t = (T_1t * W_{11} + T_2t * W_{01}) *$ denom;

compute $q_t = (T_2t * W_{00} + T_1t * W_{01}) *$ denom; and

determine a second gradient vector (G_t) that identifies the direction of maximum change in a second texture coordinate (t) as $G_t = p_t * D_1 + q_t * D_2$.

33. (Previously Presented) The method of claim 32 further comprising:

determining, for each vertex of the at least one of the graphics primitives, a 3D coordinate frame formed by a surface normal vector, tangent vector, and binormal vector, where the tangent vector is determined as directed in the general direction of the *s* texture coordinate gradient vector (G_s) determined for the at least one graphics primitive and where the binormal vector is determined as directed in the general direction of the *t* texture coordinate gradient vector (G_t) determined for the at least one graphics primitive.

34. (New) The method of claim 1 wherein D_1 is a vector from a first of the plurality of vertices of the graphics primitive for which the at least two texture coordinate gradient vectors is being computed to a second of the plurality of vertices of the graphics primitive, and wherein said computing at least two texture coordinate gradient vectors further comprises:

computing a second dot product (W_{01}) of a second vector (D_2) with D_1, wherein D_2 is a vector from the first of the plurality of vertices of the graphics primitive for which the at least two texture coordinate gradient vectors is being computed to a third of the plurality of vertices of the graphics primitive;

computing a third dot product (W_{11}) of D_2 with itself;

assigning to a variable a value computed based at least in part on W_{00}, W_{01}, and W_{11}; and

utilizing said variable as a denominator in calculating linear combination variables that are used in further computing said at least two texture coordinate gradient vectors.

35. (New) The method of claim 34 wherein said assigning to a variable a value computed based at least in part on W_{00}, W_{01}, and W_{11} comprises:

assigning said variable a value computed as a result of $1/(W_{00} * W_{11} - W_{01} * W_{01})$.

36. (New) The method of claim 28 wherein said first vector is a vector from one of the plurality of vertices of the graphics primitive for which the at least two texture coordinate gradient vectors are being computed to another of the plurality of vertices of the graphics primitive for which the at least two texture coordinate gradient vectors are being computed.